Julia for QCD spectroscopy

Misha Mikhasenko

July 2021

Julia in my projects

Interface between theory and experiment needs non-standard functions, flexibility

$\sim$ 2017: Mathematica / C++ $\Rightarrow$ Julia

- [**JPAC**, PRD 98 (2018)]: Studies of the $a_1(1260)$
  - Coding complex parametrization
  - Binned fit with `NLopt`.
  - Analytic continuation
- [**COMPASS**, PRL 127 (2021) 8, 082501]: Triangle Singularity as the Origin of the $a_1(1420)$
  - Coding complex parametrization (cross check for C++)
  - Final plotting
- [**LHCb**, PRD Lett., 2107.03419]: Ω_c^{**0} in Ω_b^+ decays
  - Event processing: cut-bases selection
  - Event-based log-likelihood fitting
- [**LHCb**, submitted to Nature Physics]: Study of the doubly charmed tetraquark T_{cc}^+
  - Coding complex parametrization (cross check for C++)
  - Analytic studies: integrals, complex algebra

Workflow in LHCb analysis

Workflow:

1. Course selection on event topology: `python/DSL` to configure C++ algorithms of `Gaudi`
2. Pre-Selection (done via distributed computation grid): `python/DSL` to configure C++
3. Fine event selection – NN/BDT: `python/C++`
4. Extraction of physics: `python/C++`

The steps 3 and 4 can be done by Julia.

The reason why it is not done:

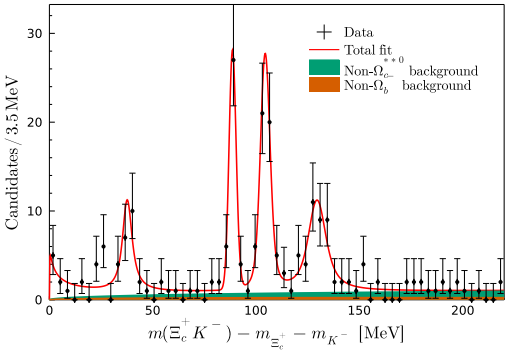
- Code inheritance: many standard steps (BDTs) across many analysis
- Number of non-C++ precedents is very low

PDF builder and parameter handler in Julia

Replace `RooFit`

Construction complex parametric PDFs:

- Non-standard densities (complex interferences $|A + B|^2$)
- Non-linear dependence on parameters
- Parameter constraints
- Unbinned (extended) maximum likelihood fitting



[https://github.com/mmikhasenko/AlgebraPDF.jl]

Minimizers

Replace `Minuit`

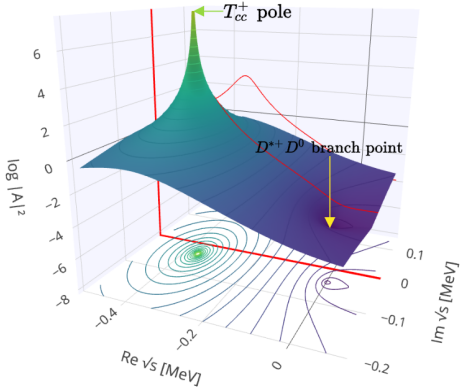
Optimizers in Julia:

- `Optim.jl`: implement `BFGS`
  - Suggests a common interface of minimization: `NLSolversBase`
- `NLopt.jl`: calls `BFGS`
  - C++ wrapper, works
- `iminuit` via `PyCall.jl`: calls `MIGRAD`, pre-version of `BFGS`
  - Incredibly stable linear search, works where other (above) fail
  - `iminuit` 1.54 $\rightarrow$ 2.7.0: versioning via `Conda.jl` is not straightforward

Plotting in Julia

Replace ROOT

- GR via `Plots.jl`
  - Super fast, publication quality
  - Convenient recipe system
  - Relatively easy to implement hacks
  - Might be unstable, misses hep-related recipes
- `PGFplotsX.jl`: LaTeX plots
- `PlotlyJS.jl`: interactive



[Interactive plot in cloud by Plotly Chart Studio.]

Summary

Julia is ready to be used in physics HEP analysis

What will make experience even better:

- ROOT format reading without python: `UpROOT`
- Minimizers: finding stopping criteria, tuning linear search
- ...

More enthusiasts pioneering application of Julia!